МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

**КУРСОВИЙ ПРОЕКТ**

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: «Об’єктно-орієнтоване програмування»

на тему:

**«ВІДДІЛ КАДРІВ»**

студента І курсу групи ІПЗ-20-1

спеціальності 121 «Інженерія програмного забезпечення»

Кравченко Володимир Олександрович

(прізвище, ім’я та по-батькові)

Керівник ст.викладач кафедри ІПЗ Чижмотря О.В.,

Дата захисту: " 11" вересня 2023 р.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Житомир – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Факультет інформаційно-комп’ютерних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень: бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

А.В.Морозов

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ р.

ЗАВДАННЯ

НА КУРСОВИЙ ПРОЕКТ СТУДЕНТУ

Кравченко Володимир Олександрович

1. Тема роботи: Розробка програми «Відділ кадрів»,

керівник роботи: ст.в. Чижмотря О.В.

1. Строк подання студентом: “25” жовтня 2023р.
2. Вихідні дані до роботи: Розробити програму "Відділ кадрів".
3. Зміст розрахунково-пояснювальної записки(перелік питань. Які підлягають розробці)
   * + 1. Постановка завдання
       2. Аналіз аналогічних розробок
       3. Алгоритми роботи програми

4. Опис роботи програми

5. Програмне дослідження

1. Перелік графічного матеріалу(з точним зазначенням обов’язкових креслень)

1.Презентація до КП

2.Посилання на репозиторій:

1. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посади консультанта | Підпис, дата | |
| завдання  видав | завдання прийняв |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Дата видачі завдання “ 22 ” лютого 2021 р.

|  |  |  |  |
| --- | --- | --- | --- |
| № з/п | Назва етапів курсового проекту | Строк виконання етапів проекту | Примітки |
| 1 | Постановка задачі | 6.05.2021 |  |
| 2 | Пошук, огляд та аналіз аналогічних розробок | 8.08.2021 |  |
| 3 | Формулювання технічного завдання | 10.08.2021 |  |
| 4 | Опрацювання літературних джерел | 15.08.2021 |  |
| 5 | Проектування структури | 20.08.2021 |  |
| 6 | Написання програмного коду | 22.09.2021 |  |
| 7 | Відлагодження | 2.10.2023 |  |
| 8 | Написання пояснювальної записки | 15.10.2023 |  |
| 9 | Захист | 25.10.2023 |  |

**КАЛЕНДАРНИЙ ПЛАН**

**Студент** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кравченко В. О.

(підпис) (прізвище та ініціали)

**Керівник проекту** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чижмотря О. В.

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Завданням на курсовой проект (роботу) було створення програми для збереження, редагування, створення, видалення, пошуку персоналу.

Пояснювальна записка до курсового проекту (роботи) на тему «Відділ кадрів» складається з вступу, трьох розділів, висновків, списку використаної літератури та додатків.

Текстова частина викладена на 21 сторінках друкованого тексту.

Пояснювальна записка має 17 сторінок додатків. Список використаних джерел містить 6 найменувань і займає 1 сторінку. В роботі наведено 19 рисунків. Загальний обсяг роботи – 38 сторінок.

Ключові слова: ДОДАТОК, ПРОГРАМА, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ.

ЗМІСТ

ВСТУП………………………………………………………………………………7

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ…………………………………………………………………….………..7

* 1. Аналіз задачі, засобів та методів її вирішення………………………7
  2. Аналіз існуючого програмного забезпечення за тематикою курсового проекту…………………………………………………….........................7

Висновки до першого розділу…………………………………….…………10

РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....................................................................................................11

2.1 Проектування загального алгоритму роботи програми……………...11

2.2 Розробка функціональних алгоритмів роботи програми…………….12

2.3 Розробка програмного забезпечення…………………………………..13

Висновки до розділу………………………………………………………...14

РОЗДІЛ 3. ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ…….................................................................................................15

3.1 Опис роботи з програмним додатком…………………………………15

3.2 Тестування роботи програмного забезпечення……………………….24

Висновок до розділу………………………………………………………...24

ВИСНОВКИ…………………………………………………………………….......25

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ………………………………………….26

ДОДАТКИ………………………………………………………………………......27

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ:

ОС – Операційна система

ПЗ – Програмне забезпечення

ТЗ – Технічне завдання

ПКМ – права клавіша миши

Рис – рисунок

БД - база данних

І-п.с - інформаційно-пошукові системи

Б-ка – бібліотека

**РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ**

**Аналіз задачі, засобів та методів її вирішення**

Задача полягає у тому, що потрібно:

1) Обрати потрібну інформацію

2) Визначити потреби звичайного користувача

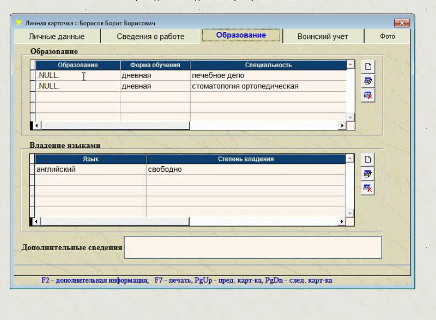
3) Створити простий інтерфейс програми В процесі створення даних брався аспект повних даних.

Було вирішено розробити додаток у Visual Studio за технологією WinForm, оскільки він має широкі можливості в створенні функціоналу програми.

**Аналіз існуючого програмного забезпечення за тематикою курсової роботи**.

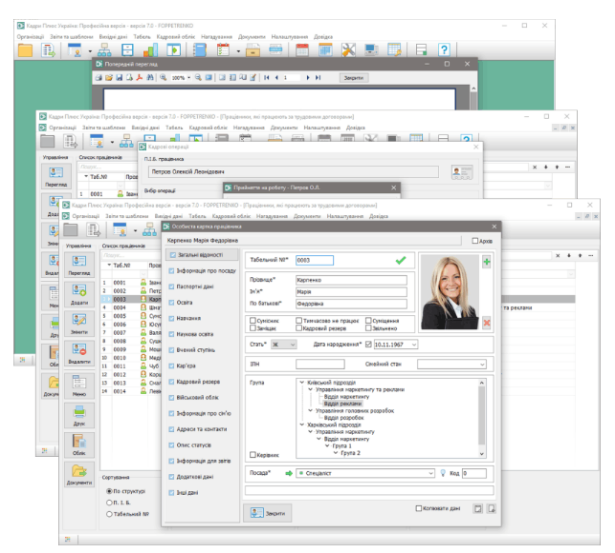
Для аналізу існуючого програмного забезпечення за тематикою курсового проекту, було вибрано три конкуруючі програми:

**ЧижКадри** – програма кадрового обліку для організацій будь-якої форми власності та виду діяльності і в той же час дуже проста в роботі. Користувач може налаштувати дані і функції, необхідні саме для нього. Створена для організації кадрів всередині програмного продукту.



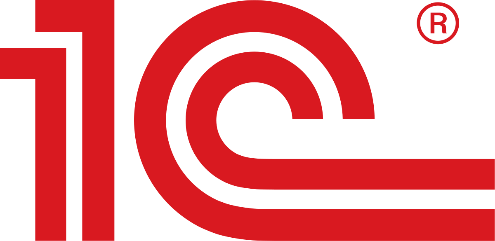
**Рис.1.1.** ЧижКадри

**Кадри Плюс Україна** – платна програма для кадрового обліку. За допомогою **"Кадри Плюс"** можна з легкістю створювати накази, заяви, звіти, інші типові документи, відслідковувати рух персоналу і вести облік робочого часу. Програма також має інші додаткові функції, які будуть корисні кадровому працівнику.



**Рис.1.2.** Кадри Плюс Україна

**1С Кадри** – Кадровий облік в програмі 1С. Російська компанія, що спеціалізується на дистрибуції, підтримці і розробці комп'ютерних програм і баз даних ділового та домашнього призначення.



**Рис.1.3.** 1С

**Підстава для розробки:**

Підставою для розробки додатку стало завдання курсового проектування

**Призначення розробки:**

Програма призначення для використання користувачем для адміністрування відділом кадрів.

**Вимоги до програмного додатку:**

Програма повинна надати зберігати, редагувати, видаляти, захищати, записувати працівників введені користувачем на персональному компютері.

**Програма повинна виконувати наступні функції:**

1. Виведення всієї необхідної інформації.

2. Редагування кадрів

3. Відображення кадрів

4. Зберігання кадрів

Потрібно притримуватись максимальної простоти інтерфейсу.

У разі виконання некоректних дій передбачено підказки та приклад максимальне роз’яснення причини, через яку певна дія не вдалась чи вдалась некоректно.

**Вимоги до надійності:**

Передбачити ситуації при виконанні некоректних дій з боку користувача.

**Вимоги до ПК:** Система повинна працювати на IBM-сумісних комп’ютерах та під управлінням ОС Windows.

**Вимоги до інтерфейсу**: Інтерфейс головного вікна додатку повинен виконувати функції додання людини до списку користувачів бібліотеки,а також передбачена ситуація, коли користувач вже є у системі.

**Спеціальні вимоги:** Немає.

**Висновки до першого розділу**

У ході виконання першого розділу було визначено дозволені можливості користувача та зону роботи, можливість доступу до інформаціях з різними рівнями безпеки. Було визначено необхідні засоби та інструменти для реалізації програмного забезпечення, а також інтерфейсу користувача, яку необхідно створити.

**РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

**2.1 Проектування загального алгоритму роботи програми**

На даному етапі необхідно створити загальний алгоритм роботи програмного додатку. Після входу до програми користувача зустрічатиме вікном з повним списком усіх записів. Запис має у собі наступні данні:

1. Список кадрів

2. Поле додавання працівника

3. Поле редагування працівника

4. Поле видалення позицій

5. Коротка інформація про кожного працівника

На рисунку 2.1 зоображена загальна блок схема роботи Програмного Додатка

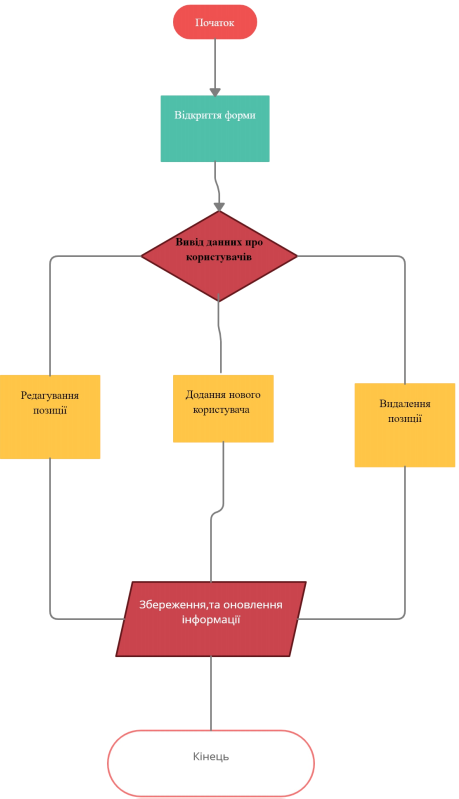


Рис 2.1. Загальний алгоритм

**2.2 Розробка функціональних алгоритмів роботи програми**

Користувач має можливість створення нових кадрів, редагування. Для створення таких завдань створювались різні алгоритми переглянути які можна у додатках.

**2.3 Розробка програмного забезпечення**

Після завершення проектування можна перейти до розробки програмного забезпечення. Етап проектування спростив розробку додатку та дав поштовх для розробки програми на відповідні класи та вікна для розробки програми. Було створено рішення та проект, який складається з форм та БД. Кожне вікно та клас вирішує свою низку проблем та завдань.

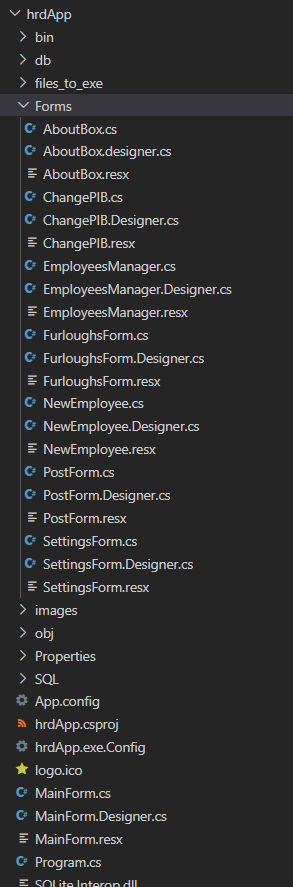


Рис 2.2 Вигляд структури рішення

На рисунку 2.2 можна побачити файлову структуру рішення програмного додатку. У файлі MainForm.cs зберігається розмітка головного вікна, повний код файлу, можна знайти розмітку у додатку Б. Розмітку вікна можна побачити у рисунку 2.3.

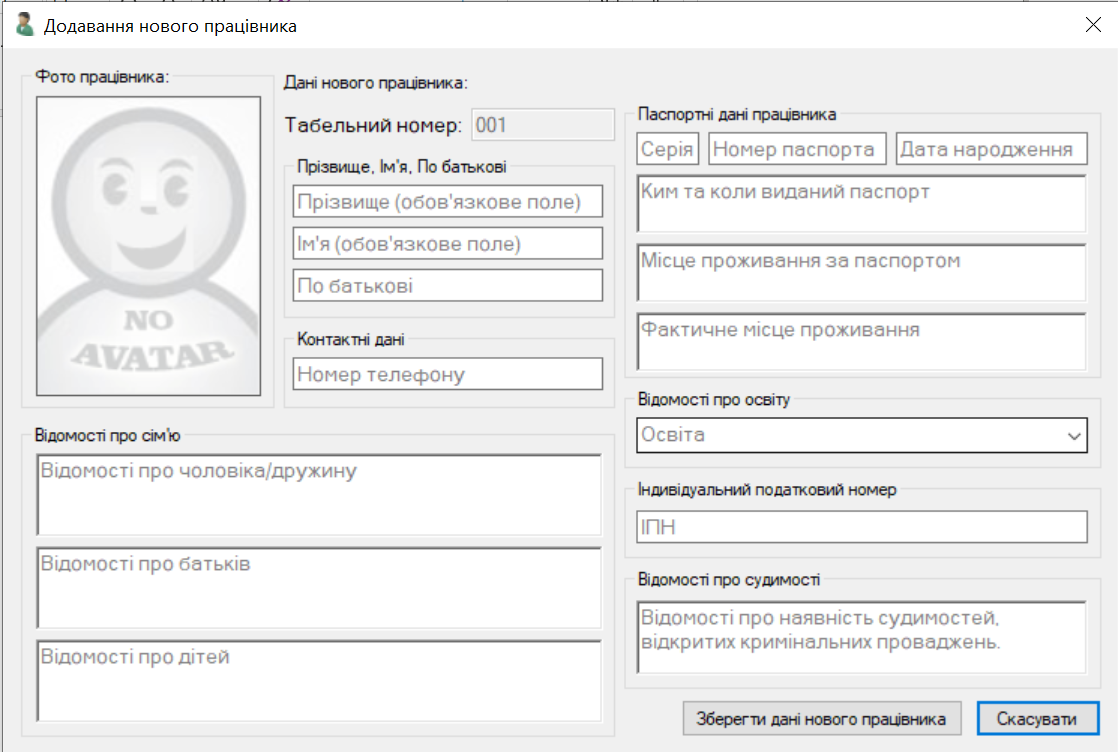


Рис 2.3. Вигляд форми додавання

**Висновки до другого розділу**

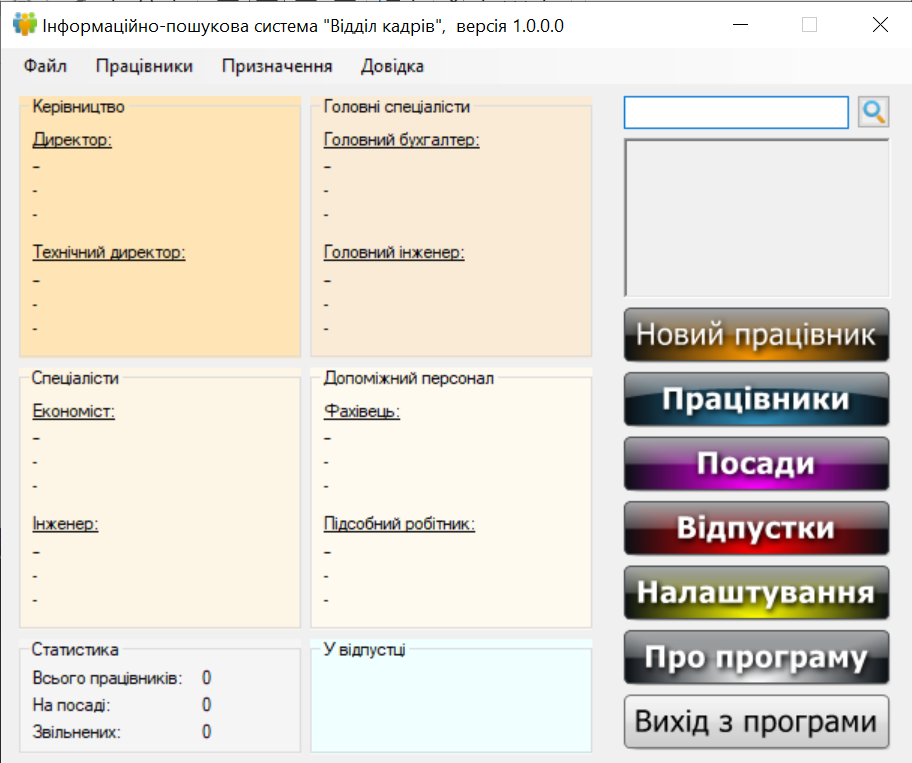
У ході виконання другого розділу було спроектовано та розроблено у вигляді блок-схеми загальний алгоритм, розроблено програмний продукт на основі блок-схеми.

**РОЗДІЛ 3. ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО**

**ТЕСТУВАННЯ**

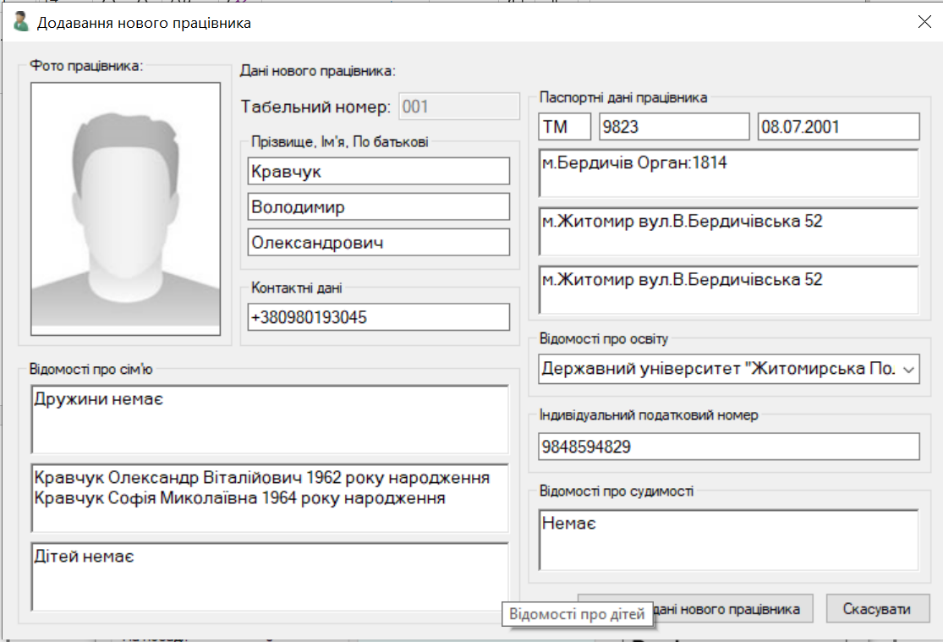
**3.1 Опис роботи з програмним додатком**

Для запуску програмного додатку завантажуємо програму й відкриваємо exe-файл “Інформаційно-пошукова система «Відділ кадрів» v.1.0.0.0.exe”. Відкривається головне меню, в якому є вся навігація (кнопки створення, посад, відпусток, налаштування, допомоги, тощо…):

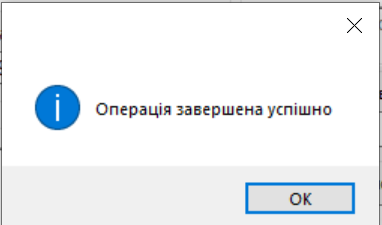


**Рис.3.1** Головне меню програми

Додамо нового працівника до програми:

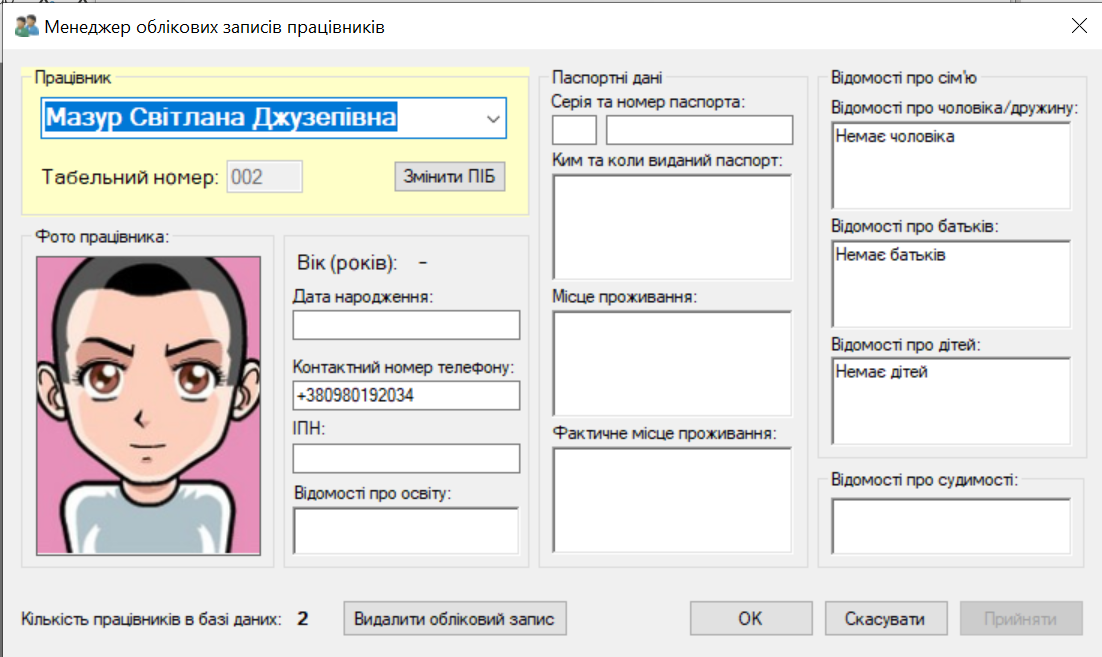


**Рис.3.2** Вікно додавання працівника



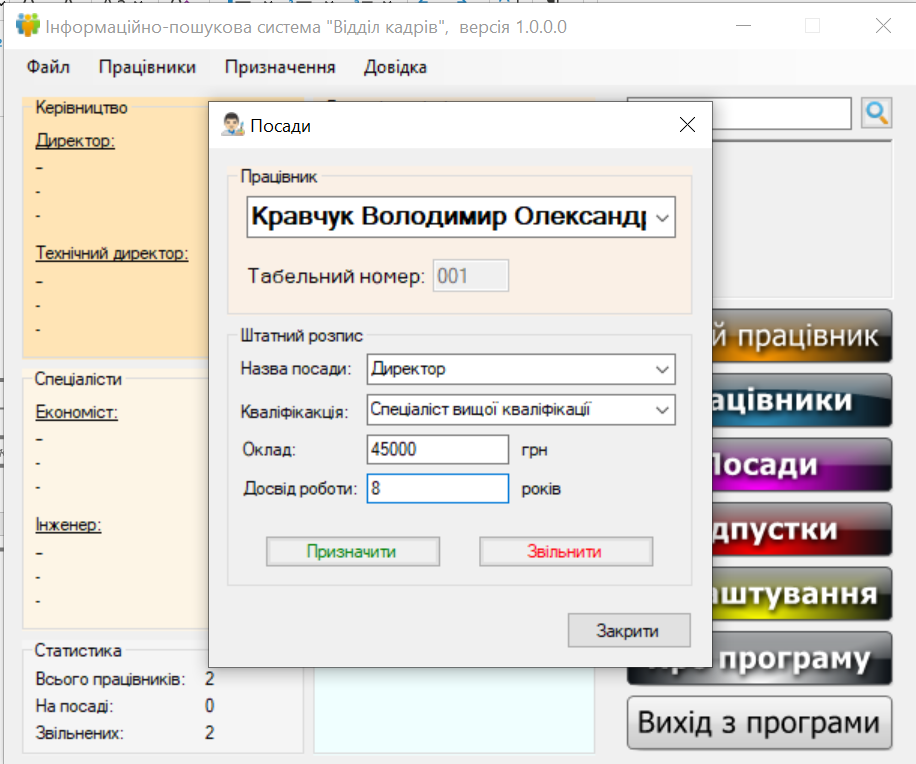
**Рис.3.2** Вікно підтвердження

Додамо пару працівників й переглянемо їх картки:

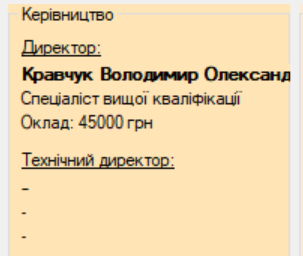


**Рис.3.3** Вікно працівників

Призначемо посаду директора Володимиру:

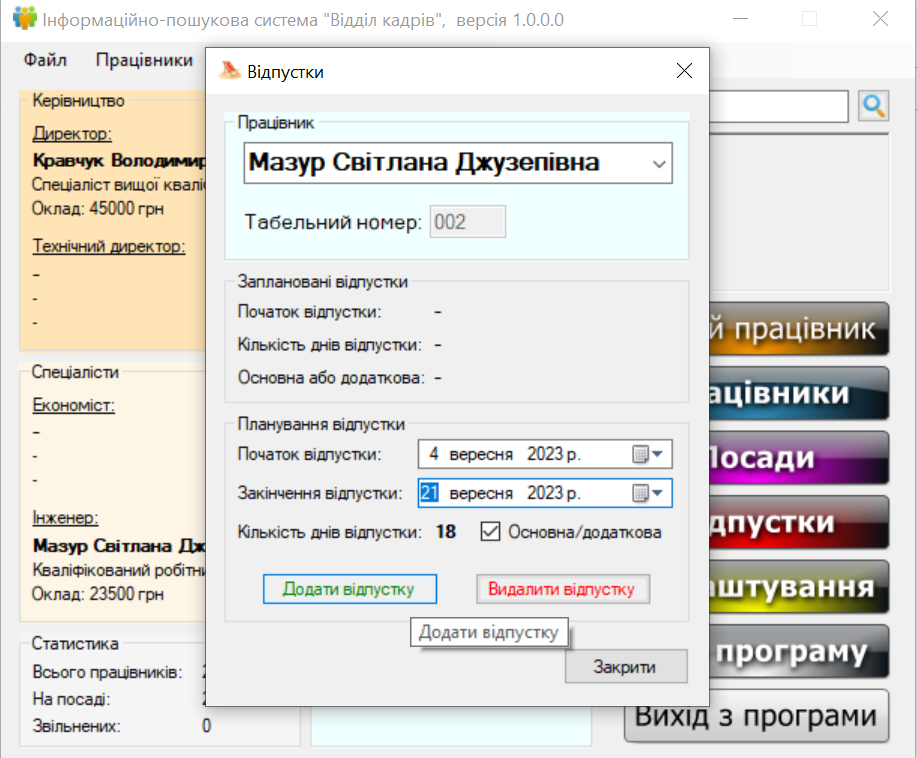


**Рис.3.4** Призначення посади директора



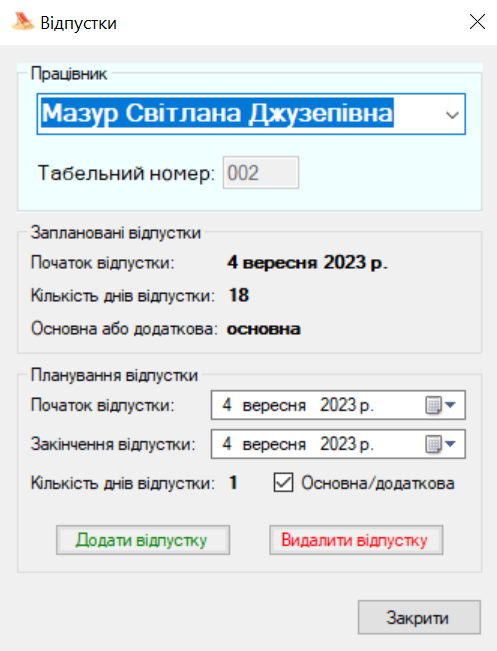
**Рис.3.5** Результат

Призначемо відпустку одній із співробітниць:



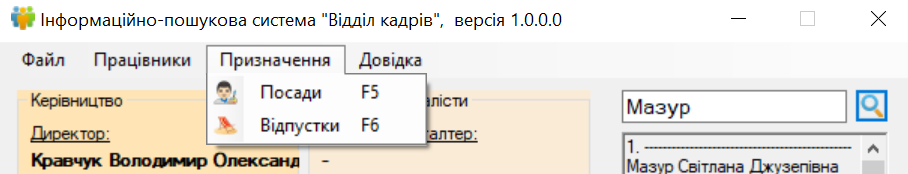
**Рис.3.6** Призначення відпустки

Переглянемо результат, що поля заповнені. Також є можливість видалення відпустки чи оновлення їх данних:



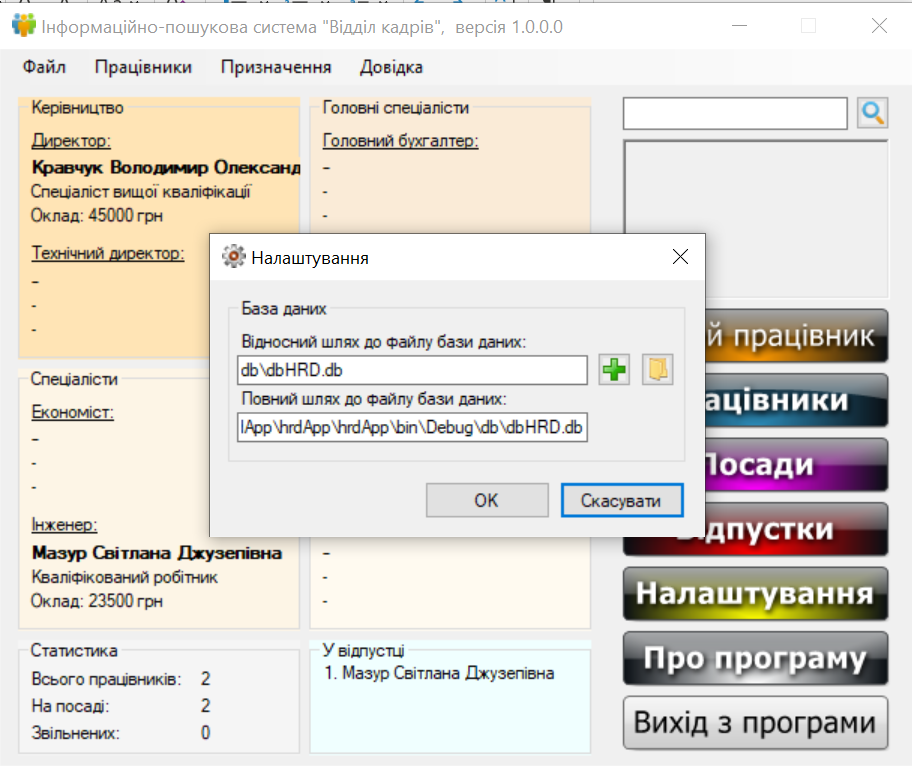
**Рис.3.7** Результат

У функціоналі програми передбачена швидка панель керування:



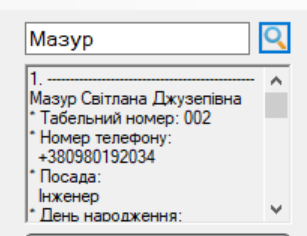
**Рис.3.8** Швидка панель

Також програма має певні налаштування, а саме призначення місця збереження бази данних:



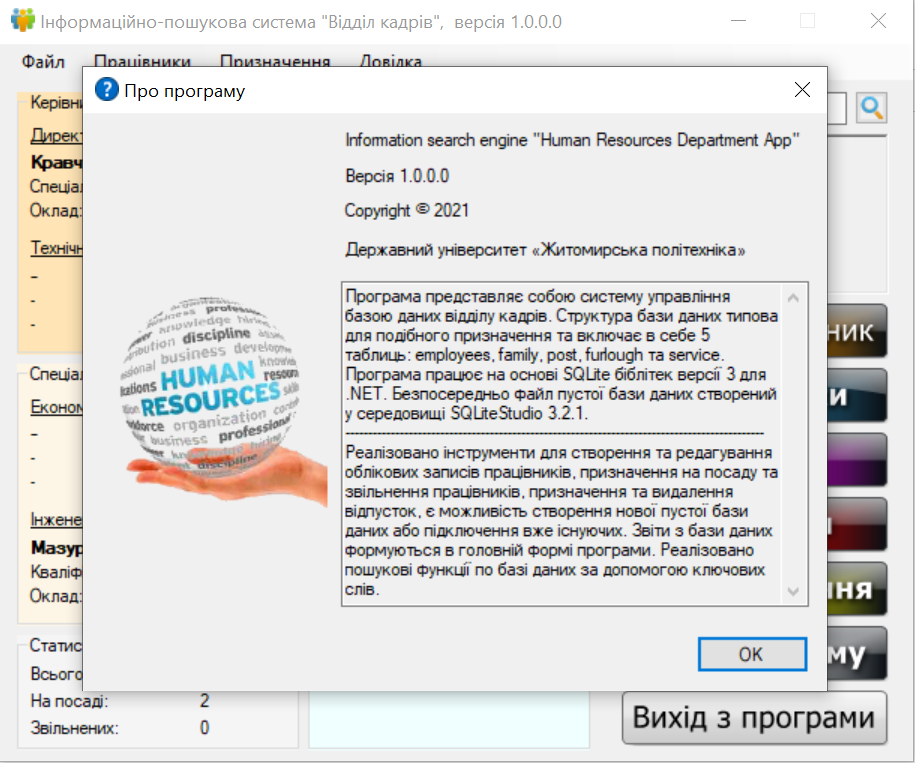
**Рис.3.9** Налаштування

Програма має функціонал пошуку:



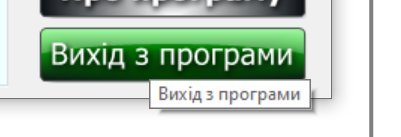
**Рис.4.0** Пошук

Програма має певну інформацію, яка доступна для перегляду нашому користувачеві, так звана довідка:



**Рис.4.0** Налаштування

Щоб вийти із програми й автоматично зберегти всі данні, скористаємось спеціальною кнопкою “Вихід з програми”:



**Рис.4.1** Вихід із програми

**3.2. Тестування роботи програмного забезпечення**

Було прийняте рішення проведення ручного тестування. Та після підтвердження відсутності помилок, було прийнято рішення, що помилок немає або вони не впливають на роботу. Однак якщо користувач введе не відповідні значення його про це попередить программа

**Висновок до третього розділу**

В цьому розділі була описана документація до розробленого додатку, яке описує роботу з програмним додатком для звичайного користувача. Програма має загалом інтуїтивно зрозумілий інтерфейс.

В результаті тестування були виявлені та виправлені помилки, після виправляння помилок, тестування було проведено повторно. На даному етапі робота програми є стійкою до збоїв.

Загалом у процесі тестування було засвідчено, що програма виконує свої функції рівно так, як від неї очікувалось

**ВИСНОВКИ**

В результаті виконання цієї роботи вивчено основи прикладної галузі, також отримав досвід у написанні програми ”Відділ кадрів”. На перший погляд тема є дуже простою та що в програмі немає нічного складного, але в процесі реалізації постало завдання, зробити все в найкращому вигляді, автоматично та з передбаченням можливих помилок. Було проведено тестування у ручному режимі, після виявлення помилок, їх було виправлено та пере-протестовано додаток, що дає можливість визнати, що додаток є стабільним. Таким чином була розроблена програма з використанням мови C#, .NET, технології Windows Form, та досліджена галузь збереження та обробки інформації.

**Список використаних джерел**

1.Уроки C# .NET Windows Forms. [Електронний ресурс] / Гоша Дударь // YouTube. – 2019. – Режим доступу до ресурсу: <https://www.youtube.com/watch?v=uXlMHzjBEDU>

2. Шилдт Г. ПОЛНОЕ РУКОВОДСТВО C# 4.0 / Гереберт Шилдт. – Москва: Издательский дом "Вильямс", 2011. – 1056 с. – (MC Graw hill). – (ББК; кн. 32973).

3. metanit. metanit [Електронний ресурс] / metanit // metanit.com. – 2012. – Режим доступу до ресурсу: <https://metanit.com/sharp/adonet/2.5.php>

4. Microsoft. Microsoft C# documentation [Електронний ресурс] / Microsoft // Microsoft Build. – 2020. – Режим доступу до ресурсу: <https://docs.microsoft.com/en-us/dotnet/csharp/>.

5. stackoverflow [Електронний ресурс] / stackoverflow // https://ru.stackoverflow.com – 2012. – Режим доступу до ресурсу: <https://ru.stackoverflow.com/?tags=c%23>

6. coderoad [Електронний ресурс] / coderoad // https://coderoad.ru – 2012. – Режим доступу до ресурсу:https://coderoad.ru/25266591/как-установить- базу-данных-с-приложением-Windows-Form

**ДОДАТКИ**

Додаток А

Вихідний код проекту

**PostForm.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace hrdApp

{

    public partial class PostForm : Form

    {

        private Boolean fst\_change\_Salary;

        private Boolean fst\_change\_WorkExperience;

        public static int number\_of\_employee;

        public static Boolean isNewData;

        public static string RegNumber\_DB;

        public static string Post\_DB;

        public static string Qualification\_DB;

        public static string Salary\_DB;

        public static string WorkExperience\_DB;

        public PostForm()

        {

            InitializeComponent();

        }

        private void b\_Back\_Click(object sender, EventArgs e)

        {

            Close();

        }

        private void b\_Back\_KeyDown(object sender, KeyEventArgs e)

        {

            if (e.KeyCode == Keys.Escape)

            {

                Close();

            }

        }

        private void PostForm\_Shown(object sender, EventArgs e)

        {

            MainForm.UpdateCountEmployee\_from\_DB();

            UpdateEmployees();

            ReloadData();

            isNewData = false;

            fst\_change\_Salary = true;

            fst\_change\_WorkExperience = true;

        }

        private void UpdateEmployees()

        {

            // тимчасова зупинка обробників переривань для cb\_Employee

            this.cb\_Employee.TextChanged -= new System.EventHandler(this.cb\_Employee\_TextChanged);

            this.cb\_Employee.SelectedIndexChanged -= new System.EventHandler(this.cb\_Employee\_SelectedIndexChanged);

            ///////////////////////////////////////////////////////////////////////////////////////////////

            if (MainForm.N\_Employees > 0)

            {

                cb\_Employee.Items.Clear();

                for (int i = 0; i < MainForm.N\_Employees; i++)

                {

                    string name = MainForm.Employee\_LastName[i] + " "

                                + MainForm.Employee\_FirstName[i] + " "

                                + MainForm.Employee\_Surname[i];

                    if (name != null)

                        cb\_Employee.Items.Add(name);

                }

                cb\_Employee.SelectedIndex = 0;

                number\_of\_employee = cb\_Employee.SelectedIndex;

                tb\_RegNumber.Text = MainForm.Employee\_RegNumber[number\_of\_employee];

            }

            // відновлення обробників переривань для cb\_Employee

            this.cb\_Employee.TextChanged += new System.EventHandler(this.cb\_Employee\_TextChanged);

            this.cb\_Employee.SelectedIndexChanged += new System.EventHandler(this.cb\_Employee\_SelectedIndexChanged);

            ///////////////////////////////////////////////////////////////////////////////////////////////

        }

        private void ReloadData()

        {

            // тимчасова зупинка обробників переривань

            this.cb\_Employee.TextChanged -= new System.EventHandler(this.cb\_Employee\_TextChanged);

            //////////////////////////////////////////////////////////////////////////////////////////////////

            if (MainForm.N\_Employees > 0)

            {

                MainForm.UpdatePost\_from\_DB(tb\_RegNumber.Text);

                //

                cb\_Post.Text = MainForm.Post;

                cb\_Qualification.Text = MainForm.Qualification;

                tb\_Salary.Text = MainForm.Salary;

                tb\_WorkExperience.Text = MainForm.WorkExperience;

            }

            else

            {

                cb\_Employee.Text = null;

                tb\_RegNumber.Text = null;

            }

            // відновлення обробників переривань

            this.cb\_Employee.TextChanged += new System.EventHandler(this.cb\_Employee\_TextChanged);

            ///////////////////////////////////////////////////////////////////////////////////////////////

        }

        private void cb\_Employee\_SelectedIndexChanged(object sender, EventArgs e)

        {

            if (cb\_Employee.SelectedIndex >= 0)

            {

                number\_of\_employee = cb\_Employee.SelectedIndex;

                tb\_RegNumber.Text = MainForm.Employee\_RegNumber[number\_of\_employee];

                ReloadData();

            }

        }

        private void cb\_Employee\_TextChanged(object sender, EventArgs e)

        {

        }

        public static bool IsNumeric(object Expression)

        {

            double retNum;

            bool isNum = Double.TryParse(Convert.ToString(Expression), System.Globalization.NumberStyles.Any, System.Globalization.NumberFormatInfo.InvariantInfo, out retNum);

            return isNum;

        }

        private void StartRegNewPost()

        {

            RegNumber\_DB = tb\_RegNumber.Text;

            Post\_DB = cb\_Post.Text;

            Qualification\_DB = cb\_Qualification.Text;

            Salary\_DB = tb\_Salary.Text;

            WorkExperience\_DB = tb\_WorkExperience.Text;

            if (cb\_Post.Text != "" && cb\_Qualification.Text != "" && IsNumeric(tb\_Salary.Text) && IsNumeric(tb\_WorkExperience.Text))

                isNewData = true;

            else

                MessageBox.Show("Потрібно заповнити всі поля штатного розпису!", "", MessageBoxButtons.OK, MessageBoxIcon.Warning);

        }

        private void b\_AddPost\_Click(object sender, EventArgs e)

        {

            StartRegNewPost();

            MainForm.CreateNewPost();

            ReloadData();

            fst\_change\_Salary = true;

            fst\_change\_WorkExperience = true;

        }

        private void b\_Close\_Click(object sender, EventArgs e)

        {

            Close();

        }

        private void b\_Close\_KeyDown(object sender, KeyEventArgs e)

        {

            if (e.KeyCode == Keys.Escape)

            {

                Close();

            }

        }

        private void tb\_Salary\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_Salary && tb\_Salary.Text == "")

            {

                fst\_change\_Salary = false;

                tb\_Salary.Text = "1000";

            }

        }

        private void tb\_WorkExperience\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_WorkExperience && tb\_WorkExperience.Text == "")

            {

                fst\_change\_WorkExperience = false;

                tb\_WorkExperience.Text = "0";

            }

        }

        private void b\_DeleteFurlough\_Click(object sender, EventArgs e)

        {

            if (MainForm.N\_Employees > 0)

            {

                if (MessageBox.Show("Ви дійсно бажаєте звільнити працівника? \r\n" +

                                    "Дану операцію скасувати буде неможливо.", "", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)

                {

                    MainForm.DeletePost\_from\_DB(MainForm.Employee\_RegNumber[number\_of\_employee]);

                    UpdateEmployees();

                    ReloadData();

                    fst\_change\_Salary = true;

                    fst\_change\_WorkExperience = true;

                }

            }

            else

            {

                MessageBox.Show("База даних пуста. Звільнення неможливе.", "", MessageBoxButtons.OK, MessageBoxIcon.Warning);

            }

        }

        private void b\_AddPost\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(b\_AddPost, "Призначити працівника на вказану посаду");

        }

        private void b\_DeleteFurlough\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(b\_DeleteFurlough, "Звільнити працівника із вказаної посади");

        }

    }

}

**NewEmployee.cs**

using System;

using System.Drawing;

using System.IO;

using System.Globalization;

using System.Windows.Forms;

namespace hrdApp

{

    public partial class NewEmployeeForm : Form

    {

        private Boolean fst\_change\_RegNumber;

        private Boolean fst\_change\_LastName;

        private Boolean fst\_change\_FirstName;

        private Boolean fst\_change\_Surname;

        private Boolean fst\_change\_PassportSeries;

        private Boolean fst\_change\_PassportNumber;

        private Boolean fst\_change\_AboutPassport;

        private Boolean fst\_change\_ITN;

        private Boolean fst\_change\_BirthDay;

        private Boolean fst\_change\_Residence;

        private Boolean fst\_change\_ActualResidence;

        private Boolean fst\_change\_Phone;

        private Boolean fst\_change\_Education;

        private Boolean fst\_change\_AboutManWoman;

        private Boolean fst\_change\_AboutParents;

        private Boolean fst\_change\_AboutChildren;

        private Boolean fst\_change\_Conviction;

        public static string img\_path\_photo = MainForm.img\_photo\_path\_def;

        public static string img\_no\_photo = "no\_photo";

        public static Boolean isNewData;

        public static string RegNumber\_DB;

        public static string LastName\_DB;

        public static string FirstName\_DB;

        public static string Surname\_DB;

        public static string PassportSeries\_DB;

        public static string PassportNumber\_DB;

        public static string AboutPassport\_DB;

        public static string ITN\_DB;

        public static string BirthDay\_DB;

        public static string Residence\_DB;

        public static string ActualResidence\_DB;

        public static string Phone\_DB;

        public static string Education\_DB;

        public static string AboutManWoman\_DB;

        public static string AboutParents\_DB;

        public static string AboutChildren\_DB;

        public static string Conviction\_DB;

        public static string photo\_path\_DB;

        static SQL.SQLClass sql\_ne = new SQL.SQLClass();

        public static int MaxNewEmployeesInDB = -1;

        public NewEmployeeForm()

        {

            InitializeComponent();

        }

        private void b\_OK\_Click(object sender, EventArgs e)

        {

            StartRegNewEmployee();

            MainForm.CreateNewEmployee();

        }

        private void b\_Cancel\_Click(object sender, EventArgs e)

        {

            Close();

        }

        private void b\_Cancel\_KeyDown(object sender, KeyEventArgs e)

        {

            if (e.KeyCode == Keys.Escape)

            {

                Close();

            }

        }

        private void tb\_RegName\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_RegNumber)

            {

                fst\_change\_RegNumber = false;

                tb\_RegNumber.Text = null;

                tb\_RegNumber.ForeColor = Color.Black;

            }

        }

        private void tb\_LastName\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_LastName)

            {

                fst\_change\_LastName = false;

                tb\_LastName.Text = null;

                tb\_LastName.ForeColor = Color.Black;

            }

        }

        private void tb\_FirstName\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_FirstName)

            {

                fst\_change\_FirstName = false;

                tb\_FirstName.Text = null;

                tb\_FirstName.ForeColor = Color.Black;

            }

        }

        private void tb\_Surname\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_Surname)

            {

                fst\_change\_Surname = false;

                tb\_Surname.Text = null;

                tb\_Surname.ForeColor = Color.Black;

            }

        }

        private void tb\_PassportSeries\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_PassportSeries)

            {

                fst\_change\_PassportSeries = false;

                tb\_PassportSeries.Text = null;

                tb\_PassportSeries.ForeColor = Color.Black;

            }

        }

        private void tb\_PassportNumber\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_PassportNumber)

            {

                fst\_change\_PassportNumber = false;

                tb\_PassportNumber.Text = null;

                tb\_PassportNumber.ForeColor = Color.Black;

            }

        }

        private void tb\_ITN\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_ITN)

            {

                fst\_change\_ITN = false;

                tb\_ITN.Text = null;

                tb\_ITN.ForeColor = Color.Black;

            }

        }

        private void tb\_BirthDay\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_BirthDay)

            {

                fst\_change\_BirthDay = false;

                tb\_BirthDay.Text = MainForm.dateFormat;

                tb\_BirthDay.ForeColor = Color.Black;

            }

        }

        private void rtb\_Residence\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_Residence)

            {

                fst\_change\_Residence = false;

                rtb\_Residence.Text = null;

                rtb\_Residence.ForeColor = Color.Black;

            }

        }

        private void tb\_Phone\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_Phone)

            {

                fst\_change\_Phone = false;

                tb\_Phone.Text = "+380";

                tb\_Phone.ForeColor = Color.Black;

            }

        }

        private void cb\_Education\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_Education)

            {

                fst\_change\_Education = false;

                cb\_Education.Text = null;

                cb\_Education.ForeColor = Color.Black;

            }

        }

        private void rtb\_ActualResidence\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_ActualResidence)

            {

                fst\_change\_ActualResidence = false;

                rtb\_ActualResidence.Text = null;

                rtb\_ActualResidence.ForeColor = Color.Black;

            }

        }

        private void rtb\_AboutManWoman\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_AboutManWoman)

            {

                fst\_change\_AboutManWoman = false;

                rtb\_AboutManWoman.Text = "";

                rtb\_AboutManWoman.ForeColor = Color.Black;

            }

        }

        private void rtb\_AboutParents\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_AboutParents)

            {

                fst\_change\_AboutParents = false;

                rtb\_AboutParents.Text = "";

                rtb\_AboutParents.ForeColor = Color.Black;

            }

        }

        private void rtb\_AboutChildren\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_AboutChildren)

            {

                fst\_change\_AboutChildren = false;

                rtb\_AboutChildren.Text = "";

                rtb\_AboutChildren.ForeColor = Color.Black;

            }

        }

        private void rtb\_Conviction\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_Conviction)

            {

                fst\_change\_Conviction = false;

                rtb\_Conviction.Text = "";

                rtb\_Conviction.ForeColor = Color.Black;

            }

        }

        private void rtb\_AboutPassport\_Enter(object sender, EventArgs e)

        {

            if (fst\_change\_AboutPassport)

            {

                fst\_change\_AboutPassport = false;

                rtb\_AboutPassport.Text = null;

                rtb\_AboutPassport.ForeColor = Color.Black;

            }

        }

        private void rtb\_Residence\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_Residence, "Місце проживання за паспортом");

        }

        private void tb\_RegName\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_RegNumber, "Табельний номер працівника");

        }

        private void tb\_LastName\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_LastName, "Прізвище працівника");

        }

        private void tb\_FirstName\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_FirstName, "Ім'я працівника");

        }

        private void tb\_Surname\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_Surname, "По батькові працівника");

        }

        private void tb\_PassportSeries\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_PassportSeries, "Серія паспорта");

        }

        private void tb\_PassportNumber\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_PassportNumber, "Номер паспорта");

        }

        private void rtb\_AboutPassport\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_AboutPassport, "Ким та коли виданий паспорт");

        }

        private void tb\_ITN\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_ITN, "Індивідуальний Податковий Номер");

        }

        private void tb\_BirthDay\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_BirthDay, "Дата народження");

        }

        private void tb\_Phone\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(tb\_Phone, "Номер телефону");

        }

        private void pB\_user\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(pB\_user, "Додати фото працівника");

        }

        private void rtb\_ActualResidence\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_ActualResidence, "Фактичне місце проживання");

        }

        private void cb\_Education\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(cb\_Education, "Дані про освіту працівника");

        }

        private void rtb\_AboutManWoman\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_AboutManWoman, "Відомості про чоловіка/дружину");

        }

        private void rtb\_AboutParents\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_AboutParents, "Відомості про батьків");

        }

        private void rtb\_AboutChildren\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_AboutChildren, "Відомості про дітей");

        }

        private void rtb\_Conviction\_MouseHover(object sender, EventArgs e)

        {

            toolTip.SetToolTip(rtb\_Conviction, "Відомості про наявність судимостей, відкритих кримінальних проваджень.");

        }

        private void pB\_user\_Click(object sender, EventArgs e)

        {

            openFileDialog\_photo.InitialDirectory = Path.Combine(Directory.GetCurrentDirectory(), MainForm.img\_photo\_path\_def);

            openFileDialog\_photo.FileName = null;

            if (openFileDialog\_photo.ShowDialog() == DialogResult.OK)

            {

                img\_path\_photo = openFileDialog\_photo.FileName;

                pB\_user.Image = Image.FromFile(img\_path\_photo);

            }

        }

        public static string ConvertFilePath(string orig\_path, string dir\_name)

        {

            int seek\_IndexOf = orig\_path.IndexOf(dir\_name);

            if (seek\_IndexOf > 0)

                return orig\_path.Substring(seek\_IndexOf).Replace("\\", "/");

            else if (seek\_IndexOf == 0)

                return "repeat";

            else

                return null;

        }

        public static string CopyAndConvertFilePath(string orig\_path, string dir\_name)

        {

            string new\_path = Path.Combine(dir\_name, Path.GetFileName(orig\_path));

            try

            {

                File.Copy(orig\_path, new\_path, true);

            }

            catch

            {

                MessageBox.Show("Сталася помилка під час копіювання графічного файлу до папки з програмою.", "", MessageBoxButtons.OK, MessageBoxIcon.Error);

            }

            return new\_path;

        }

        private void StartRegNewEmployee()

        {

            if (ConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_def) != "repeat" &&

                ConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_usr) != "repeat" &&

                img\_path\_photo != MainForm.img\_path\_photo\_def)

            {

                if (ConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_def) != null)

                    img\_path\_photo = ConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_def);

                else if (ConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_usr) != null)

                    img\_path\_photo = ConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_usr);

                else

                    img\_path\_photo = CopyAndConvertFilePath(img\_path\_photo, MainForm.img\_photo\_path\_usr);

            }

            RegNumber\_DB = tb\_RegNumber.Text;

            if (!fst\_change\_LastName) LastName\_DB = tb\_LastName.Text; else LastName\_DB = null;

            if (!fst\_change\_FirstName) FirstName\_DB = tb\_FirstName.Text; else FirstName\_DB = null;

            if (!fst\_change\_Surname) Surname\_DB = tb\_Surname.Text; else Surname\_DB = null;

            if (!fst\_change\_PassportSeries) PassportSeries\_DB = tb\_PassportSeries.Text; else PassportSeries\_DB = null;

            if (!fst\_change\_PassportNumber) PassportNumber\_DB = tb\_PassportNumber.Text; else PassportNumber\_DB = null;

            if (!fst\_change\_AboutPassport) AboutPassport\_DB = rtb\_AboutPassport.Text; else AboutPassport\_DB = null;

            if (!fst\_change\_ITN) ITN\_DB = tb\_ITN.Text; else ITN\_DB = null;

            if (!fst\_change\_Residence) Residence\_DB = rtb\_Residence.Text; else Residence\_DB = null;

            if (!fst\_change\_ActualResidence) ActualResidence\_DB = rtb\_ActualResidence.Text;  else ActualResidence\_DB = null;

            if (!fst\_change\_Phone) Phone\_DB = tb\_Phone.Text; else Phone\_DB = null;

            if (!fst\_change\_Education) Education\_DB = cb\_Education.Text; else  Education\_DB = null;

            if (!fst\_change\_AboutManWoman) AboutManWoman\_DB = rtb\_AboutManWoman.Text; else AboutManWoman\_DB = null;

            if (!fst\_change\_AboutParents) AboutParents\_DB = rtb\_AboutParents.Text; else AboutParents\_DB = null;

            if (!fst\_change\_AboutChildren) AboutChildren\_DB = rtb\_AboutChildren.Text; else AboutChildren\_DB = null;

            if (!fst\_change\_Conviction) Conviction\_DB = rtb\_Conviction.Text; else Conviction\_DB = null;

            if (img\_path\_photo != MainForm.img\_photo\_path\_def) photo\_path\_DB = img\_path\_photo; else  photo\_path\_DB = img\_no\_photo;

            if (!fst\_change\_BirthDay && tb\_BirthDay.Text != "")

            {

                BirthDay\_DB = DateTime.ParseExact(tb\_BirthDay.Text, MainForm.dateFormat, CultureInfo.InvariantCulture).ToShortDateString();

            }

            else

                BirthDay\_DB = null;

            if ( (!fst\_change\_LastName && LastName\_DB != "") && (!fst\_change\_FirstName && FirstName\_DB != "") )

            {

                isNewData = true;

                Close();

            }

            else

            {

                MessageBox.Show("Не заповнені обов'язкові поля!", "", MessageBoxButtons.OK, MessageBoxIcon.Warning);

            }

        }

        private void NewEmployeeForm\_Shown(object sender, EventArgs e)

        {

            fst\_change\_RegNumber = true;

            fst\_change\_LastName = true;

            fst\_change\_FirstName = true;

            fst\_change\_Surname = true;

            fst\_change\_PassportSeries = true;

            fst\_change\_PassportNumber = true;

            fst\_change\_AboutPassport = true;

            fst\_change\_ITN = true;

            fst\_change\_BirthDay = true;

            fst\_change\_Residence = true;

            fst\_change\_ActualResidence = true;

            fst\_change\_Phone = true;

            fst\_change\_Education = true;

            fst\_change\_AboutManWoman = true;

            fst\_change\_AboutParents = true;

            fst\_change\_AboutChildren = true;

            fst\_change\_Conviction = true;

            isNewData = false;

            img\_path\_photo = MainForm.img\_path\_photo\_def;

            pB\_user.Image = Image.FromFile(img\_path\_photo);

            sql\_ne.conn\_db\_Update();

            if (!sql\_ne.OpenDB())

                MessageBox.Show("Помилка відкривання бази даних!", "", MessageBoxButtons.OK, MessageBoxIcon.Error);

            else

            {

                MaxNewEmployeesInDB = sql\_ne.GetMaxRegNumber();

                if (!sql\_ne.CloseDB())

                    MessageBox.Show("Помилка закривання бази даних!", "", MessageBoxButtons.OK, MessageBoxIcon.Error);

            }

            tb\_RegNumber.Text = (MaxNewEmployeesInDB + 1).ToString("000");

        }

    }

}